(19) Europäisches Patentamt
European Patent Office
Office européen des brevets

(11) EP 1 388 782 A1

(12) EUROPEAN PATENT APPLICATION

(43) Date of publication:
11.02.2004 Bulletin 2004/07

(51) Int Cl.[7]: **G06F 9/44**

(21) Application number: **02009955.2**

(22) Date of filing: **03.05.2002**

(84) Designated Contracting States:
AT BE CH CY DE DK ES FI FR GB GR IE IT LI LU MC NL PT SE TR
Designated Extension States:
AL LT LV MK RO SI

(71) Applicant: **SAP Aktiengesellschaft**
69190 Walldorf (DE)

(72) Inventor: **Stienhans, Frank**
69168 Wiesloch (DE)

(54) **Method and computer system for providing stateful favorites**

(57) Method, system and computer program product for providing stateful favorites. A client (900) creates a stateful favorite (250) by sending a request for a reduced state (210) of a stateful application session (200-S) to a server (901). The client receives the reduced state (210) from the server (901) and stores (430) the reduced state (210) as a portion of the stateful favorite (250).

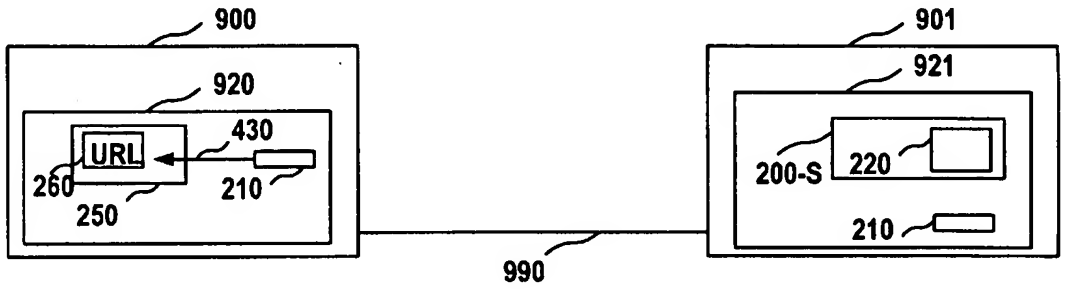

FIG. 3C

EP 1 388 782 A1

Description

Field of the Invention

[0001] The present invention generally relates to electronic data processing, and more particularly, relates to method, computer program product and system for improved user interaction.

Background of the Invention

[0002] Favorites, as provided for example by conventional browsers, such as the Microsoft Internet explorer, are known by those skilled in the art. When a user accesses an application through a browser on a client, the user can use an "Add-Favorite"-function to add a representation (Favorite) of the application to a "Favorites-Menu" of the browser. Later on the user can use the representation to reload the application into the browser. If the application is a stateless application, in other words, if the current state of the application is available on the client,. the current state of the application can be saved by the browser when the user adds the corresponding representation to the "Favorites-Menu". When the user returns to the application through the "Favorites-Menu", the state of the application is restored.

[0003] World Wide Web (WWW) concepts focus on stateless scenarios, where the client knows the complete state of the application. For many business related applications, stateless applications show insufficient performance because large amounts of data have to be transmitted from the server to the client to make all state information available on the client. Further, data traffic created by stateless business applications requires high-bandwidth communication lines between the server and the client. The "Add-Favorite"-function, as described above, does not support "Favorites" related to stateful applications, that is, applications that store the state on the server, where the application is running, rather than on the client.

Summary of the Invention

[0004] Therefore, it is an objective of the present invention to provide methods, computer program products and computer systems to provide client-side favorites in a browser, wherein the favorites relate to stateful applications.

[0005] To meet this objective, methods according to claims 1, 5 and 8 are provided. Further embodiments of the present invention provide a computer system running computer programs to execute the methods.

[0006] It is an effect of the present invention that a stateful favorite for a stateful application on a server is defined by a client (for example through a user of the client) without the need of favorite-information to be stored on the server. Therefore, management of stateful favorites can be handled completely by the client without the risk of leaving obsolete favorite-information on the server when, for example, deleting a stateful favorite on the client.

[0007] It is a further effect of the present invention that the client does not need to store a complete session-state with the favorite in the browser to restore the session-state on the server. Because the session-state of a stateful application can reach several mega bytes in size, extracting mandatory information from the session-state saves bandwidth in the client-server communication and further saves memory for storing the favorite on the client.

[0008] It is a further effect of the present invention that a stateful application behaves like a stateless application when using a corresponding stateful favorite in the browser of the client. That is, when creating a stateful favorite, a reduced state of the stateful application is transmitted from the server to the client. When executing the stateful favorite the reduced state is transmitted to the server similar to a stateless application.

[0009] The aspects of the invention will be realized and attained by means of the elements and combinations particularly pointed out in the appended claims. It is to be understood that both, the foregoing general description and the following detailed description are exemplary and explanatory only and are not restrictive of the invention as described.

Brief Description of the Drawings

[0010]

FIG. 1	illustrates a simplified block diagram of a computer system that can be used with an embodiment of the present invention;
FIGS. 2A-2D	illustrate a graphical user-interface of a stateful application session at different time points;
FIGS. 3A-3C	illustrate a client and a server when creating a stateful favorite in one embodiment of the present invention;
FIG. 4	illustrates details of extracting a reduced state from a complete state;
FIGS. 5A-5C	illustrate the client and the server when restoring a stateful application session in one embodiment of the present invention;

FIG. 6 illustrates details of extending the reduced state to the complete state; and
FIGS. 7A-7C illustrate simplified flow charts of methods to handle stateful favorites according to the present invention.

Detailed Description of the Invention

[0011] Wherever possible, the same reference numbers will be used throughout the drawings to refer to the same or like parts. For convenience of explanation a list of references is provided prior to the claims.

Definitions of terms, as used hereinafter:

[0012] client:
client computer in a computer system.
[0013] server:
server computer in the computer system.
[0014] session:
a session is an application instance running on a server for a specific user.
[0015] session-state:
a session-state includes session dependent information, such as identifiers of objects used by an application. The session dependent information provides an environment to run an application instance of the application on a server. For convenience of explanation the "session-state" is referred to as "state", hereinafter.
[0016] reduced state:
portion of a state that allows to restore the complete state of a session.
[0017] stateful application:
a stateful application holds its state on the server, where the corresponding application instance is running.
[0018] stateless application:
a stateless application does not hold its state on the server, where the corresponding application instance is running. The complete state of a stateless application is hold on the client side.
[0019] stateful favorite:
a stateful favorite is a shortcut to an application, that can be used by a user to restore a state of a session on a server by launching the stateful favorite from the client. Stateful favorites can have a long lifetime and typically are valid even after a session ends. For convenience of explanation, "stateful favorites" are referred to as "favorites", hereinafter. Favorites can be included in the "Favorites-Menu" of a browser or made available as a shortcut on the client (e.g. via an icon or a specific key-sequence).
[0020] serialize:
serialize, as used hereinafter, means transforming data into a serialized data format.
[0021] FIG. 1 illustrates a simplified block diagram of an exemplary computer system, that implements embodiments of the present invention. Computer system 999 includes computer 900 and has a plurality of further computers 901, 902 (or 90q, with q=0...Q-1, Q any number).
[0022] Computer 900 can be connected to further computers 901, 902 via computer network 990. Computer 900 comprises processor 910, memory 920, bus 930, and, optionally, input device 940 and output device 950 (I/O devices, user interface 960). As illustrated, the invention is present by computer program product 100 (CPP), program carrier 970 and program signal 980, collectively "program".
[0023] In respect to computer 900, computer 901/902 is sometimes referred to as "remote computer", computer 901/902 is, for example, a server, a router, a peer device or other common network node, and typically comprises many or all of the elements described relative to computer 900. Hence, elements 100 and 910-980 in computer 900 collectively illustrate also corresponding elements 10q and 91q-98q (shown for q=0) in computers 90q.
[0024] Computer 900 is, for example, a conventional personal computer (PC), a desktop and hand-held device, a multiprocessor computer, a pen computer, a microprocessor-based or programmable consumer electronics, a minicomputer, a mainframe computer, a personal mobile computing device, a mobile phone, a portable or stationary personal computer, a palmtop computer or the like.
[0025] Processor 910 is, for example, a central processing unit (CPU), a micro-controller unit (MCU), digital signal processor (DSP), or the like.
[0026] Memory 920 symbolizes elements that temporarily or permanently store data and instructions. Although memory 920 is conveniently illustrated as part of computer 900, memory function can also be implemented in network 990, in computers 901/902 and in processor 910 itself (e.g., cache, register), or elsewhere. Memory 920 can be a read only memory (ROM), a random access memory (RAM), or a memory with other access options. Memory 920 is physically implemented by computer-readable media, such as, for example: (a) magnetic media, such as a hard disk, a floppy

disk, or other magnetic disk, a tape, a cassette tape; (b) optical media, such as optical disk (CD-ROM, digital versatile disk - DVD); (c) semiconductor media, such as DRAM, SRAM, EPROM, EEPROM, memory stick, or by any other media, such as paper.

[0027] Optionally, memory 920 is distributed across different media. Portions of memory 920 can be removable or non-removable. For reading from media and for writing in media, computer 900 uses devices well known in the art such as, for example, disk drives, tape drives.

[0028] Memory 920 stores support modules such as, for example; a basic input output system (BIOS), an operating system (OS), a program library, a compiler, an interpreter, and a text- processing tool. Support modules are commercially available and can be installed on computer 900 by those of skill in the art. For simplicity, these modules are not illustrated.

[0029] CPP 100 comprises program instructions and-optionally - data that cause processor 910 to execute method steps of the present invention. Method steps are explained with more detail below. In other words, CPP 100 defines the operation of computer 900 and its interaction in network system 999. For example and without the intention to be limiting, CPP 100 can be available as source code in any programming language, and as object code ("binary code") in a compiled form. Persons of skill in the art can use CPP 100 in connection with any of the above support modules (e.g., compiler, interpreter, operating system).

[0030] Although CPP 100 is illustrated as being stored in memory 920, CPP 100 can be located elsewhere. CPP 100 can also be embodied in carrier 970.

[0031] Carrier 970 is illustrated outside computer 900. For communicating CPP 100 to computer 900, carrier 970 is conveniently inserted into input device 940. Carrier 970 is implemented as any computer readable medium, such as a medium largely explained above (cf. memory 920). Generally, carrier 970 is an article of manufacture comprising a computer readable medium having computer readable program code means embodied therein for executing the method of the present invention. Further, program signal 980 can also embody computer program 100. Signal 980 travels on network 990 to computer 900.

[0032] Having described CPP 100, program carrier 970, and program signal 980 in connection with computer 900 is convenient. Optionally, program carrier 971/972 (not shown) and program signal 981/982 embody computer program product (CPP) 101/102 to be executed by processor 911/912 (not shown) in computers 901/902, respectively.

[0033] Input device 940 symbolizes a device that provides data and instructions for processing by computer 900. For example, device 940 is a keyboard, a pointing device (e.g., mouse, trackball, cursor direction keys), microphone, joystick, game pad, scanner, disk drive. Although the examples are devices with human interaction, device 940 can also operate without human interaction, such as, a wireless receiver (e.g., with satellite dish or terrestrial antenna), a sensor (e.g., a thermometer), a counter (e.g., goods counter in a factory). Input device 940 can serve to read carrier 970.

[0034] Output device 950 symbolizes a device that presents instructions and data that have been processed. For example, a monitor or a display, (cathode ray tube (CRT), flat panel display, liquid crystal display (LCD), speaker, printer, plotter, vibration alert device. Similar as above, output device 950 communicates with the user, but it can also communicate with further computers.

[0035] Input device 940 and output device 950 can be combined to a single device; any device 940 and 950 can be provided optional.

[0036] Bus 930 and network 990 provide logical and physical connections by conveying instruction and data signals. While connections inside computer 900 are conveniently referred to as "bus 930", connections between computers 900-902 are referred to as "network 990". Optionally, network 990 comprises gateways being computers that specialize in data transmission and protocol conversion.

[0037] Devices 940 and 950 are coupled to computer 900 by bus 930 (as illustrated) or by network 990 (optional). While the signals inside computer 900 are mostly electrical signals, the signals in network are electrical, magnetic, optical or wireless (radio) signals.

[0038] Networking environments (as network 990) are commonplace in offices, enterprise-wide computer networks, intranets and the internet (i.e. world wide web). The physical distance between a remote computer and computer 900 is not important. Network 990 can be a wired or a wireless network. To name a few network implementations, network 990 is, for example, a local area network (LAN), a wide area network (WAN), a public switched telephone network (PSTN); a Integrated Services Digital Network (ISDN), an infra-red (IR) link, a radio link, such as Universal Mobile Telecommunications System (UMTS), Global System for Mobile Communication (GSM), Code Division Multiple Access (CDMA), or satellite link.

[0039] Transmission protocols and data formats are known, for example, as transmission control protocol/internet protocol (TCP/IP), hyper text transfer protocol (HTTP), secure HTTP, wireless application protocol, unique resource locator (URL), a unique resource identifier (URI), hyper text markup language HTML, extensible markup language (XML), extensible hyper text markup language (XHTML), wireless application markup language (WML), Standard Generalized Markup Language (SGML) etc.

[0040] Interfaces coupled between the elements are also well known in the art. For simplicity, interfaces are not

illustrated. An interface can be, for example, a serial port interface, a parallel port interface, a game port, a universal serial bus (USB) interface, an internal or external modem, a video adapter, or a sound card.

[0041] Computer and program are closely related. As used hereinafter, phrases, such as "the computer provides" and "the program provides", are convenient abbreviation to express actions by a computer that is controlled by a program.

[0042] FIGS. 2A-2D illustrate graphical user-interface 955 of a stateful application session at different time points T0, T1, T2 and T3. The following scenario describes the interaction of a user with a financial application via graphical user-interface 955. The scenario will be used as an example of one embodiment of the present invention. The present invention can be implemented in alternative embodiments for other applications. Examples for other applications are applications in the field of logistics, human capital management, customer relationship management, product life cycle management or any other application that is used by the user in a business or administrative context.

[0043] At T0 (cf. FIG. 2A), the user gets prompted with graphical user-interface 955 (GUI). For example, the user interacts with computer system 999 via user-interface 960 (cf. FIG. 1) of client 900 (cf. FIG. 1). GUI 955 is presented to the user via output device 950 (e.g. a display device such as a monitor or a PDA display) in a conventional browser. In the example, FIG. 2A illustrates the initial screen of a financial application session (also referred to as application session, hereinafter) that allows the user to query the value of a stock option. The initial screen has first screen identifier SID-1. The financial application session is running on a server, such as server 901 (cf. FIG. 1) in computer system 999. Server 901 stores the state of the session. GUI 955 includes components 955-1 to 955-4. OPTION No. component 955-1 is an entry field where the user indicates a number that identifies the option whose value is to be retrieved. VALUE component 955-2 is an output field where the retrieved option value is presented to the user. When the user starts the application session, for example by clicking on a corresponding menu entry in an application menu or on a corresponding shortcut or hyperlink on output device 950, components 955-1 and 955-2 are empty. This information is included in the state. The screen identifier SID-1 is also included in the state. SUBMIT component 955-3 and KEY FIGURES component 955-4 are components are used by the user to launch requests to the application session on server 901. For example, components 955-3, 955-4 are implemented in GUI 955 as push-buttons, shortcuts, menu entries or the like.

[0044] At T1 (cf. FIG. 2B), the user has entered a number, such as "123456", into OPTION No. 955-1. The state of the application session now includes the value "123456" of OPTION No. component 955-1. The screen identifier is still SID-1 because the layout of the screen has not changed. The user then uses SUBMIT component 955-3 to trigger the retrieval of the option value.

[0045] At T2 (cf. FIG. 2C), server 901 has retrieved the option value "2,41", which is presented to the user via VALUE component 955-2. The screen that is used to present the option value is still the screen with first screen identifier SID-1. The state of the application session now further includes the value "2,41" of VALUE component 955-2. The user uses KEY FIGURES component 955-4 to launch a request for information that further specifies option No. "123456"

[0046] At T3 (cf. FIG. 2D), server 901 has retrieved key figure values for the leverage ("8,1"), volatility ("17,2%") and spread ("0,2") of the option. The key figure values are presented to the user in LEVERAGE component 955-5, VOLATILTIY component 955-6 and SPREAD component 955-7, respectively. These components were added to GUI 955 as a result of the interaction of the user with the KEY-FIGURES component 955-4 at T2. For example, this is achieved by replacing the screen with having first screen identifier SID-1 with a further screen having second screen identifier SID-2. Therefore, the state at T3 further includes the additional components 955-5, 955-6, 955-7 and their key figure values as well as second screen identifier SID-2. In this example the user is mainly interested in the analysis of option no. "123456" because he/she is interested in buying or selling the option at the right time. Therefore, the user wants to add a new favorite-entry to the browser's "Favorites-Menu" so that, when executing the new favorite-entry, the application session as of T3 is restored and presents up-to-date values to the user. For example to achieve this, the user selects "ADD TO FAVORITES" from menu component 955-8. For example, the menu component 955-8 is part of the browser. In alternative embodiments, the user can select a corresponding button, shortcut or the like. Preferably, the user gives a name, such as "option 123456", to the new favorite-entry, which can be used later on to identify the new favorite-entry.

[0047] Client 900 then creates the new favorite-entry that stores a reduced state of the application session as of T3. The reduced state includes all information about the application session that is needed by server 901 to restore the application session at a later point in time. Therefore, the new favorite-entry is a stateful favorite according to an embodiment of the present invention. For example, restoring the application session is triggered when the user selects the new favorite-entry in the "Favorites-Menu" or when the user executes a corresponding shortcut or link.

[0048] The following figures explain further details of embodiments of the present invention with regards to the creation of the stateful favorite on client 900.

[0049] FIGS. 3A-3C illustrate client 900 and server 901 when creating stateful favorite 250 in one embodiment of the present invention. As described in FIG. 1, client 900 has memory 920 and server 901 has memory 921. Client 900 and server 901 communicate via network 990. When a first computer (e.g. client 900) sends data (e.g. reduced state

210) to a second computer (e.g. server 901) in computer system 999 (cf. FIG. 1), actually a copy of the data is sent and the original data still exists on the first computer. For convenience of explanation, the same reference number is used for the original data and the sent copy.

[0050] In the example of FIG. 3A, client 900 serves to access application session 200-S that is running on server 901 via URL 260. Application session 200-S is an instance of a stateful application. Therefore, complete state 200 of application session 200-S is stored in memory 921 of server 901. Referring back to the scenario of FIG. 2D, at T3, the user decides to create stateful favorite 250 (cf. FIG. 3C) on client 900. When selecting the "ADD TO FAVORITES" function, client 900 sends 410 request 989. Server 901 receives 510 request 981. Request 989 requests server 901 to send reduced state 210 of application session 200-S to client 900.

[0051] In FIG. 3B, server 901 extracts 520 reduced state 210 from complete state 220. In one embodiment of the present invention, extracting 520 includes the selection of relevant data of complete state 220 and the serialization of the relevant data. In another embodiment, complete state 220 is serialized and the selection of relevant data is performed using the serialized data of complete state 220. Details of the extracting 520 step are explained in FIG. 4. Server 901 then sends reduced state 210 to client 900. Upon having received 420 reduced state 210, reduced state 210 is available in memory 920. In other words, stateful application session 200-S behaves similar to a stateless application session. It is an effect of the present invention that the user does not recognize whether he/she is working with a stateless or stateful application. This results in low training efforts for the user. However, a stateless application session always makes its complete state available at client 900, whereas stateful application session 200-S makes reduced state 210 available, when it is requested by client 900. Reduced state 210 includes all information that is required by server 901 to restore application session 200-S at a later point in time.

[0052] In FIG. 3C, client 900 creates stateful favorite 250 in memory 900. For example, stateful favorite 250 can be created via the user selecting "ADD TO FAVORITES" or via dragging the URL 260 that corresponds to application session 200-S to, for example, the desktop. The URL is typically displayed in the browser. Alternative embodiments for creating stateful favorite 250 are creating a shortcut on the desktop or using a hyperlink to access the "ADD TO FAVORITES" function. That is, stateful favorite 250 can have various visual representations (e.g. menu entry, shortcut, icon, hyperlink, etc.). For example, the user or computer system 999 assigns a title to stateful favorite 250 and client 900 stores 430 reduced state 210 as a portion of stateful favorite 250. For example, stateful favorite is stored as file where the file name corresponds to the title of stateful favorite 250. Preferably, reduced state 210 is included in the file. For example, URL 260 is also included in the file.

[0053] By sending reduced state 210 instead of complete state 220 to client 900, less bandwidth is required from the client-server communication via network 990.

[0054] According to the invention reduced state 210 is sent from sever 901 to client 900 when requested. Referring back to the example of FIG. 2, a stateless application session on server 901 would transfer its current state to client 900 at each time point T0 to T3. According to an embodiment of the present invention, complete state 220 of application session 200-S is stored in memory 921 of server 901. Sending reduced state 210 to client 900 on request 989 saves further bandwidth.

[0055] According to the invention stateful favorite 250 is administrated on client 900. That is, when the user decides to delete a stateful favorite from memory 900, there is no need to notify the server, because server 901 has no knowledge of the stateful favorite on client 900.

[0056] FIG. 4 illustrates details of extracting 520 reduced state 210 from complete state 220.

[0057] Referring back to the example of FIG. 2D, at T3, complete state 220 includes information about the text of each component as illustrated in section COMPONENT TEXT. It further includes information about the value of each component as illustrated in section COMPONENT VALUE. It further includes information about the screen identifier as illustrated in section SCREEN IDENTIFIER. Further sections illustrated by ... can be included in complete state 220 to store, for example, the current position of a cursor or of a scroll bar or any other element that specifies the appearance of GUI 955 at T3.

[0058] In one embodiment of the present invention, state extractor 101-1 extracts 520 reduced state 210 from complete state 220. For example, state extractor 101-1 is a portion of computer program product 101 (cf. FIG. 1) running on server 901 to execute method 500 (cf. FIG. 7).

[0059] In the example of FIG. 2D, the information that is needed from complete state 220 to restore application session 200-S as of T3 is the value "123456" of the option number 955-1 in section COMPONENT VALUE of complete state 220 and the screen identifier SID-2 in section SCREEN IDENTIFIER of complete state 220. In the example, all texts in complete state's section COMPONENT TEXT and their corresponding display locations on the screen as well as the display locations of the values in section COMPONENT VALUE are defined on the screen having screen identifier SID-2. The values in section COMPONENT VALUE are refreshed automatically when restoring application session 200-S for option number "123456". Therefore, the value "123456" of the option number 955-1 in section COMPONENT VALUE of complete state 220 and the screen identifier SID-2 in section SCREEN IDENTIFIER of complete state 220 are included in the corresponding sections of reduced state 210. Therefore, state extractor 101-1 can be considered

as an intelligent filter of state information. In one embodiment of the present invention, state extractor 101-1 has information about the programming model of application 200 and, therefore, knows which information of a complete state of a specific application screen is essential to restore the specific application session at a later point in time. In another embodiment, state-extractor can use the object model of application 200 or any other property of application 200 that identifies data representing essential information for restoring application session 200-S. At least all the essential information is included in the corresponding reduced state.

[0060] In one embodiment of the present invention, server 901 serializes 531 reduced state 210 before sending 530 (cf. FIG. 3B) reduced state 210 to client 900. For example, serializing 531 is performed by serializer 101-2, which is an optional portion of computer program product 101.

[0061] FIGS. 5A-5C illustrate client 900 and server 901 when restoring application session 200-S in one embodiment of the present invention.

[0062] In FIG. 5A, the user selects and executes stateful favorite 250, for example by clicking on the corresponding entry in the "Favorites-Menu" of the browser on client 900 or by using one of the other representations of stateful favorite 250 that are described under FIG. 3C. Client 900 sends URL 260 to server 901. Server 901 the instantiates 610 application session 200-S in memory 921 from stateful application 200, which is also stored in server memory 921 in the example. Stateful application 200 can be stored in any storage device of computer system 999 (cf. FIG. 1). In the example, no instance (application session) of application 200 that corresponds to reduced state 210 is running on server 901 at the time when reduced state 210 is received from client 900. In case a corresponding instance (session) is already running on server 901, the instance can be replaced or a further instance can be instantiated. In one embodiment of the present invention, server 901 can prompt the user with a logon screen via client 900 if application session 200-S requires user authentication. Upon the user having logged on, server 901 continues to restore application session 200-S. If stateful application 200 supports single sign on functionality that is known by those skilled in the art the logon and authentication can be performed in the background without notifying the user.

[0063] In FIG. 5B, server 901 receives 620 reduced state 210 from client 900. Reduced state 210 is loaded into application session 200-S. In one embodiment of the present invention reduced state 210 is received as serialized data. Server 901 de-serializes the serialized data.

[0064] In FIG. 5C, server 901 extends 630 reduced state 210 into complete state 220 of application session 220-S. For example, this can be achieved by executing URL 260 with parameters that are included in reduced state 210. At this time, application session 200-S is fully restored on server 901. Details of the extending 630 step are explained in FIG. 6.

[0065] FIG. 6 illustrates details of extending 620 reduced state 210 to complete state 220.

[0066] In the example of FIG. 2D, the information that is needed from complete state 220 to restore application session 200-S as of T3 is the value "123456" of the option number 955-1 in section COMPONENT VALUE of complete state 220 and the screen identifier SID-2 in section SCREEN IDENTIFIER of complete state 220.

[0067] In one embodiment of the present invention, state extender 101-1' extends 620 reduced state 210 into complete state 220. For example, state extender 101-1' is a portion of a further computer program product 101' (cf. FIG. 1) running on server 901 to execute method 600 (cf. FIG. 7). For example, state extender 101-1' executes the application corresponding to URL 260 (cf. FIG. 5C) in application session 200-S (cf. FIG. 5C) by using the values that are included in reduced state 210 as input parameters for the application. In the example, the same values as of T3 (cf. FIG. 2D) are retrieved by the application and loaded into complete state 220 in the corresponding sections COMPONENT TEXT, COMPONENT VALUE, SCREEN IDENTIFIER an further sections...

[0068] In case, for example, the values for components 955-1, 955-2, 955-6 or 955-7 have changed since T3, the current values can be loaded into complete state 220.

[0069] In one embodiment of the present invention, server 901 de-serializes 615 reduced state 210 when receiving reduced state 210 from client 900 in a serialized data format. For example, de-serializing 615 is performed by de-serializer 101-2', which is an optional portion of computer program product 101'.

[0070] FIGS. 7A-7C illustrate simplified flow charts of methods 400, 500 and 600 to handle stateful favorites according to the present invention. Preferably, method 400 is executed by client 900 but can be executed by any other client in computer system 999 (cf. FIG. 1). For example, computer program product 100 (cf. FIG. 1) includes instructions that cause processor 910 (cf. FIG. 1) of client 900 to execute the steps of method 400. Method 400 is referred to as client method 400 in the following description. Preferably, methods 500, 600 are executed by server 901 but can be executed by any other server in computer system 999. For example, computer program product 101 (cf. FIG. 1) includes instructions that cause processor 911 (cf. FIG. 1) of server 901 to execute the steps of method 500. Method 500 is referred to as first server method 500 in the following description. For example, computer program product 101' (cf. FIG. 1) includes instructions that cause processor 911 (cf. FIG. 1) of server 901 to execute the steps of method 600. Method 600 is referred to as second server method 600 in the following description.

[0071] As shown in FIG. 7A, client method 400 for creating stateful favorite 250 on client 900 includes the following steps:

Sending 410 request 989 for reduced state 210 of stateful application session 200-S from client 900 to server 901.
Receiving 420 at client 900 reduced state 210 from server 901.
Storing 430 reduced state 210 as a portion of stateful favorite 250.
Preferably, in the receiving 420 step reduced state 210 has a serialized data format.

[0072] FIG. 7B shows a simplified flow chart of first server method 500 for providing reduced state 210 of stateful application session 200-S, wherein stateful application session 200-S has complete state 220 on server 900. First server method 500 includes the following steps:

Receiving 510 at server 901 request 989 for reduced state 210 from client 900.
Extracting 520 reduced state 210 from complete state 220.
Sending 530 reduced state 210 to client 900.
Preferably, server 901 sends reduced state 210 in a serialized data format. Therefore, optionally, server 901 serializes 525 reduced state 210 before sending 530.
FIG. 7C shows a simplified flow chart of second server method 600 for restoring stateful application session 200-S on server 901. Second server method 600 includes the following steps:

Instantiating 610 on server 901 stateful application session 200-S that corresponds to stateful favorite 250 on client 900.
Receiving 620 reduced state 210 of stateful application session 200-S on server 901, wherein reduced state 210 is a portion of stateful favorite 250 on client 900.
Extending 630 reduced state 210 to complete state 220 of stateful application session 200-S.

[0073] In one embodiment, reduced state 210 is received by server 901 in a serialized data format and optionally, server 901 de-serializes 625 reduced state 210 before extending 620.

[0074] In another embodiment of the present invention, optionally, server 901 processes 615 a logon of the user to authenticate the user. Logon procedures to authenticate a user are known by those skilled in the art.

[0075] The following coding section 1 provides a Java Script example of client-side coding that can be used on client 900 (cf. FIG. 1). Coding section 1 can be a part of an HTML-page on client 900, wherein the HTML-page represents application session 200-S running on server 901. Those skilled in the art can use coding section 1 to enable client method 400 and server methods 500, 600. A double asterisk (**) is used to indicate the start/end of comment sections in coding section 1 to explain a previous portion of the coding example for those skilled in the art.

Coding section 1:

[0076]
```
<STYLE>.sFavorite { BEHAVIOR:
    url(#default#savefavorite) } </STYLE> ** declaration for using a browser feature to locally save browser
favorite data **
<SCRIPT>
function getReducedState()
{
    var xmlhttp = new ActiveXObject("Msxml2.XMLHTTP"); xmlhttp.Open("GET", location.href +
        "?favorite=yes", false); xmlhttp.Send();
    return xmlhttp.responseXML;
} ** getReducedState() requests reduced state 210 in an XML format from server 901 ** DOM
```
[0077]
```
function fnSaveReducedState()
{
    var data=getReducedState();
    myForm.FavoriteStateControl.setAttribute
        ("sReducedState",data.xml);
}
** fnSaveReducedState() loads reduced state 210 into the variable data and stores data as attribute "sReducedState"
of stateful favorite 250 ("myForm.FavoriteState Control") **
```
[0078]
```
function fnSubmitReducedState()
{
    myForm.FavoriteStateControl.value=
    myForm.FavoriteStateControl.getAttribute
        ("sReducedState");
```

```
    myForm.submit();
}
** fnSubmitReducedState() retrieves reduced state 210 from stateful favorite 250 and sends reduced state 210 to server 901 to restore application session 200-S **
</SCRIPT> <input id="FavoriteStateControl" class="sFavorite" type="hidden" onload="fnSubmitReducedState()" onsave="fnSaveReducedState()">
** input field (that is hidden in the example but can be visible) that restores application session 200-S when stateful favorite 250 is executed (onload) and that saves reduced state 210 with stateful favorite 250 when, for example, storing stateful favorite 250 (onsave). **
```

Reference numbers	
Reference	**Description**
100/101/101'	Computer program product
200	Stateful application
200-S	Stateful application session /instance
210	Reduced state
220	Complete state
250	Stateful favorite
260	Unique resource locator (URL)
400, 500, 600	Method
4xx, 5xx, 6xx	Method steps
999	computer network system
900, 901, 902	Computers
910, 911, 912	Processors
920, 921, 922	Memories
940	Input device
950	Output device
955	Graphical user interface (GUI)
955-x	GUI components
960	User interface
970	Program carrier (computer readable device)
980	Program signal

Claims

1. A method (400) for creating a stateful favorite (250) on a client (900) comprising:

sending (410) from the client (900) to a server (901) a request (989) for a reduced state (210) of a stateful application session (200-S);
receiving (420) at the client (900) the reduced state (210) from the server (901); and
storing (430) the reduced state (210) as a portion of the stateful favorite (250).

2. The method (400) of claim 1, wherein in the receiving (420) step the reduced state (210) is received in a serialized data format.

3. The method (400) of claims 1 or 2, wherein the stateful favorite (210) has a visual representation selected from the group of menu entry, shortcut, icon and hyperlink.

4. The method (400) of claims 1, 2 or 3, wherein the stateful favorite (250) further comprises a title and a unique resource locator (260) that corresponds to the stateful application session (200-S).

5. A method (600) for restoring a stateful application session (200-S) on a server (901) comprising:

instantiating (610) on the server (901) the stateful application session (200-S) that corresponds to a stateful favorite (250) on a client (900);
receiving (620) on the server (901) a reduced state (210) of the stateful application session (200-S), wherein the reduced state (210) is a portion of the stateful favorite (250) on the client (900); and
extending (630) the reduced state (210) to a complete state (220) of the stateful application session (200-S).

6. The method (600) of claim 5, comprising:

processing (615) a logon of a user.

7. The method (600) of claims 5 or 6, wherein the reduced state (210) is received in a serialized data format and the method (600) comprises the further step:

de-serializing (625) of the reduced state (210).

8. A method (500) for providing a reduced state (210) of a stateful application session (200-S), wherein the stateful application session (200-S) has a complete state (220) on a server (901), comprising:

receiving (510) at the server (901) a request (989) for the reduced state (210) from a client (900);
extracting (520) the reduced state (210) from the complete state (220) ; and
sending (530) the reduced state (210) to the client (900).

9. The method (500) of claim 8, wherein the method (500) comprises the further step:

serializing (525) the reduced state (210).

10. A computer program product (100) for creating a stateful favorite (250) on a client (900), wherein the computer program product (100) comprises a plurality of instructions causing a processor (910) of the client (900) to execute any of the steps of claim 1.

11. A computer program product (101) for providing a reduced state (210) of a stateful application session (200-S) on a server (901), wherein the computer program product (101) comprises a plurality of instructions causing a processor (911) of the server (901) to execute any of the steps of claims 8 or 9.

12. A computer program product (101') for restoring a stateful application session (200-S) on a server (901), wherein the computer program product (101') comprises a plurality of instructions causing a processor (911) of the server (901) to execute any of the steps of claims 5, 6 or 7.

13. A computer program product (101) for providing a reduced state (210) of a stateful application session (200-S) on a server (901), wherein the computer program product (101) comprises:

a state extractor (101-1) having a plurality of instructions causing a processor (911) of the server (901) to extract (520) the reduced state (210) from a complete state (220) of the stateful application session (200-S), wherein the server (901) sends (530) the reduced state (210) to a client (900).

14. The computer program product (101) of claim 13, further comprising:

a serializer (101-1) having a plurality of instructions causing the processor (911) of the server (901) to serialize (525) the reduced state (210) before sending (530).

15. A computer program product (101') for restoring a stateful application session (200-S) on a server (901), wherein the computer program product (101') comprises:

a state extender (101-1') having a plurality of instructions causing a processor (911) of the server (901) to extend (630) a reduced state (210) of the stateful application session (200-S) to a complete state (220) of the stateful application session (200-S), wherein the server (901) instantiates (610) the stateful application session (200-S) that corresponds to a stateful favorite (250) on a client (900) and wherein the reduced state (210) is a portion of the stateful favorite (250) on the client (900).

16. The computer program product (101') of claim 15 further comprising:

a de-serializer (101-2') having a plurality of instructions causing a processor (911) of the server (901) to de-serializing (625) the reduced state (210) when received (620) from the client (900) .

17. A client (900) in a computer system (999) for creating a stateful favorite (250) on the client (900); the client (900) running a computer program (100) for executing the following steps:

sending (410) from the client (900) to a server (901) a request (989) for a reduced state (210) of a stateful application session (200-S);
receiving (420) at the client (900) the reduced state (210) from the server (901); and
storing (430) the reduced state (210) as a portion of the stateful favorite (250).

18. The client (900) of claim 17, wherein the receiving (420) step receives the reduced state (210) in a serialized data format.

19. The client (900) of claim 17, wherein the stateful favorite (210) has a visual representation selected from the group of menu entry, shortcut, icon and hyperlink.

20. The client (900) of claim 17, wherein the stateful favorite (250) further comprises a title and a unique resource locator (260) that corresponds to the stateful application session (200-S).

21. A server (901) for restoring a stateful application session (200-S); the server running a computer program (101') for executing the following steps:

instantiating (610) on the server (901) the stateful application session (200-S) that corresponds to a stateful favorite (250) on a client (900) ;
receiving (620) on the server (901) a reduced state (210) of the stateful application session (200-S), wherein the reduced state (210) is a portion of the stateful favorite (250) on the client (900); and
extending (630) the reduced state (210) to a complete state (220) of the stateful application session (200-S) .

23. The server (901) of claim 21 running the computer program (101') for executing the further step:

processing (615) a logon of a user.

24. The server (901) of claim 21, wherein the reduced state (210) is received in a serialized data format and the server (901), when running the computer program (101'), executes the further step:

de-serializing (625) of the reduced state (210).

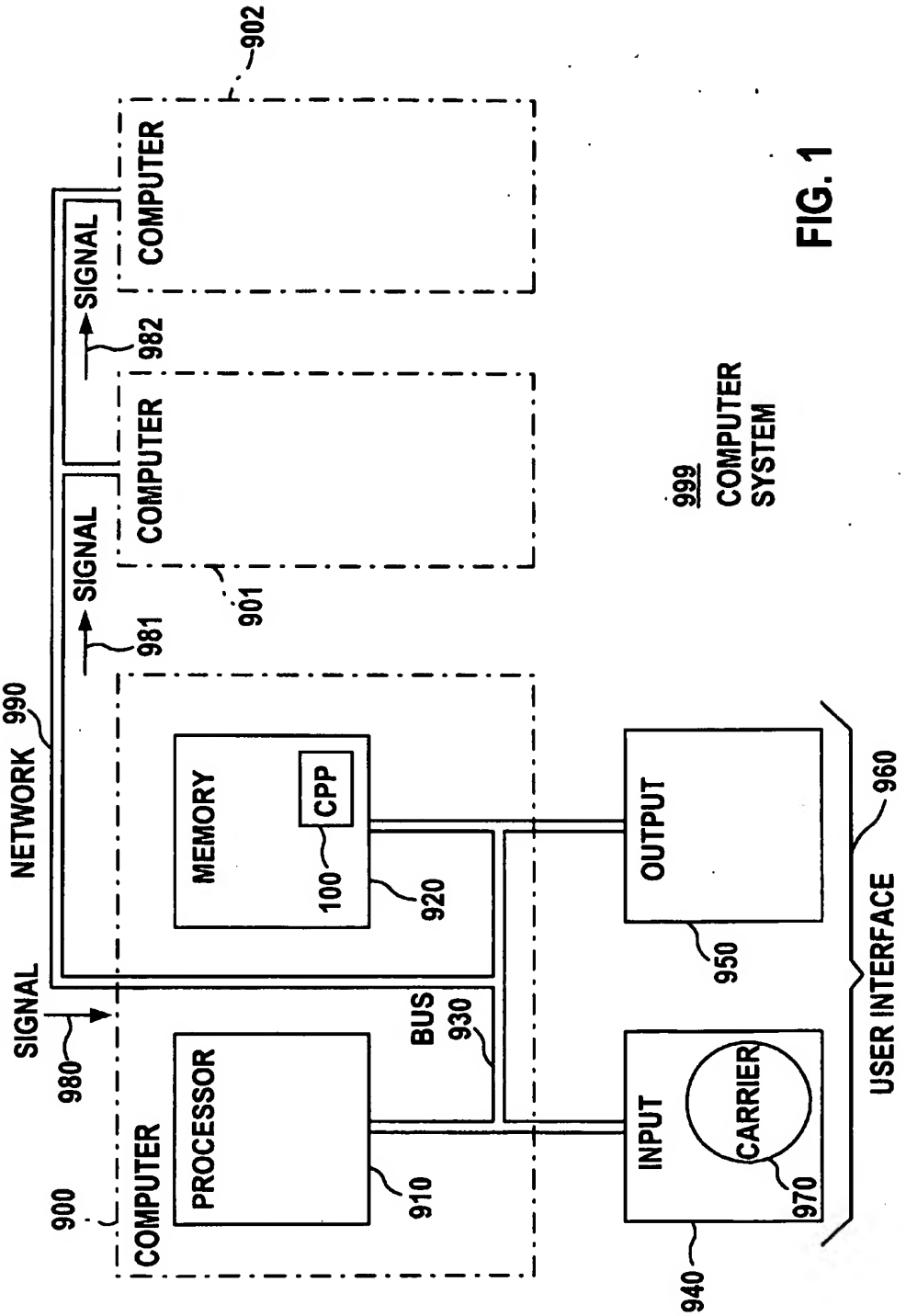
902
COMPUTER
SIGNAL
982
COMPUTER
SIGNAL
981
901
990
NETWORK
MEMORY
CPP
100
920
OUTPUT
950
960
USER INTERFACE
SIGNAL
980
BUS
930
PROCESSOR
910
INPUT
CARRIER
970
940
900
COMPUTER
999 COMPUTER SYSTEM

FIG. 1

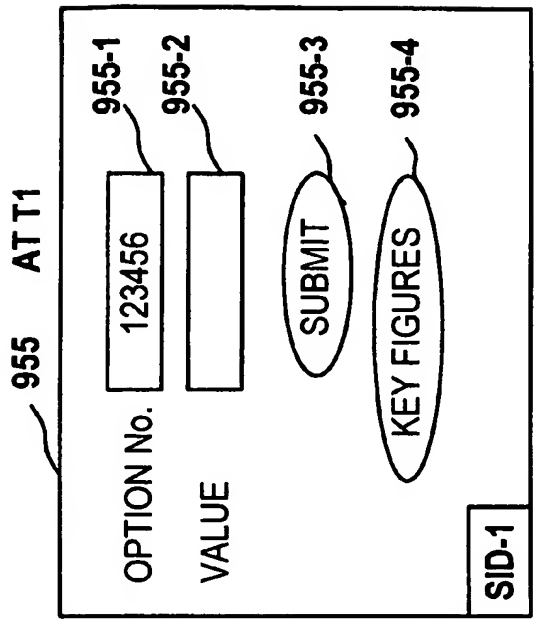
955 AT T1
955-1
955-2
955-3
955-4
OPTION No.
123456
VALUE
SUBMIT
KEY FIGURES
SID-1

FIG. 2B

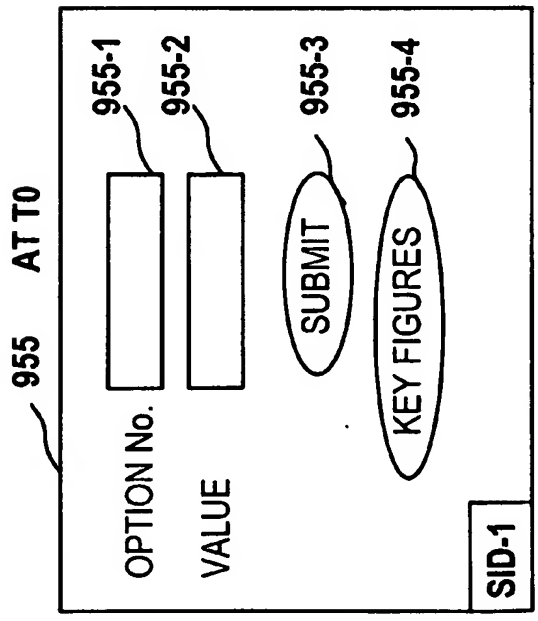
955 AT T0
955-1
955-2
955-3
955-4
OPTION No.
VALUE
SUBMIT
KEY FIGURES
SID-1

FIG. 2A

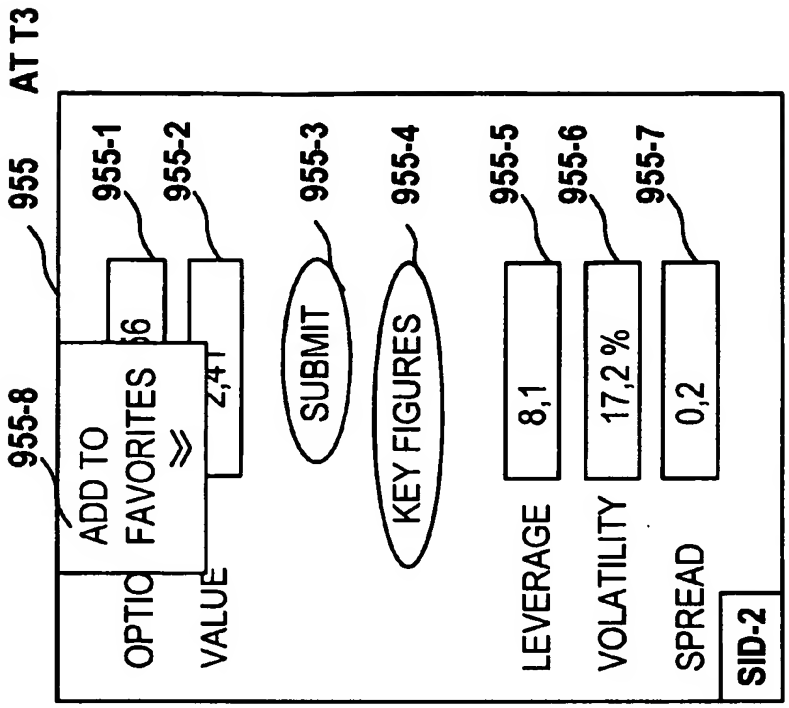
955
AT T3
955-1
955-2
955-3
955-4
955-5
955-6
955-7
955-8
ADD TO FAVORITES
OPTIO
VALUE
2,41
SUBMIT
KEY FIGURES
LEVERAGE
8,1
VOLATILITY
17,2 %
SPREAD
0,2
SID-2

FIG. 2D

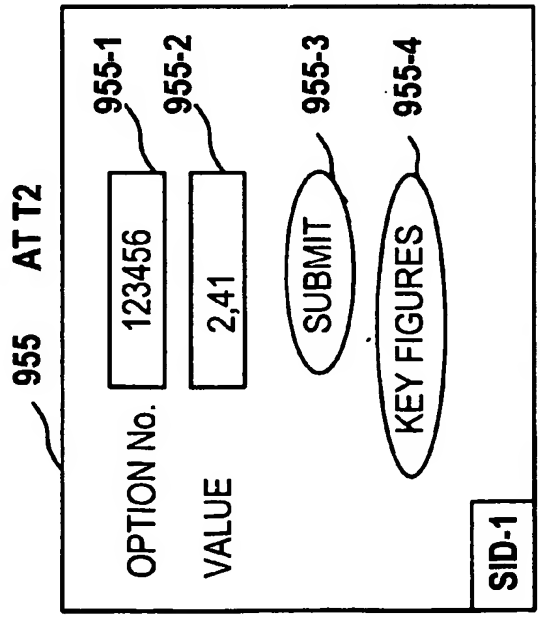
955
AT T2
955-1
955-2
955-3
955-4
OPTION No.
123456
VALUE
2,41
SUBMIT
KEY FIGURES
SID-1

FIG. 2C

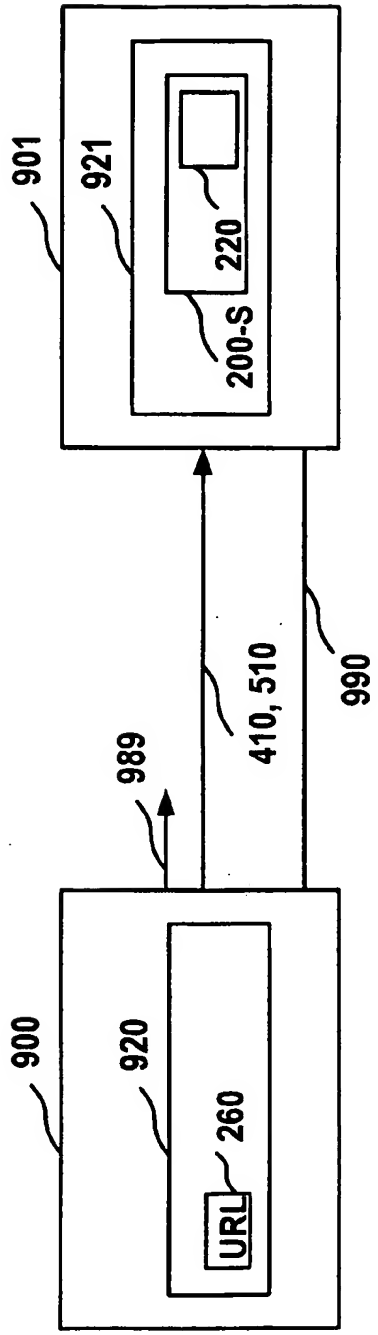
901
921
220
200-S
989
410, 510
990
900
920
260
URL

FIG. 3A

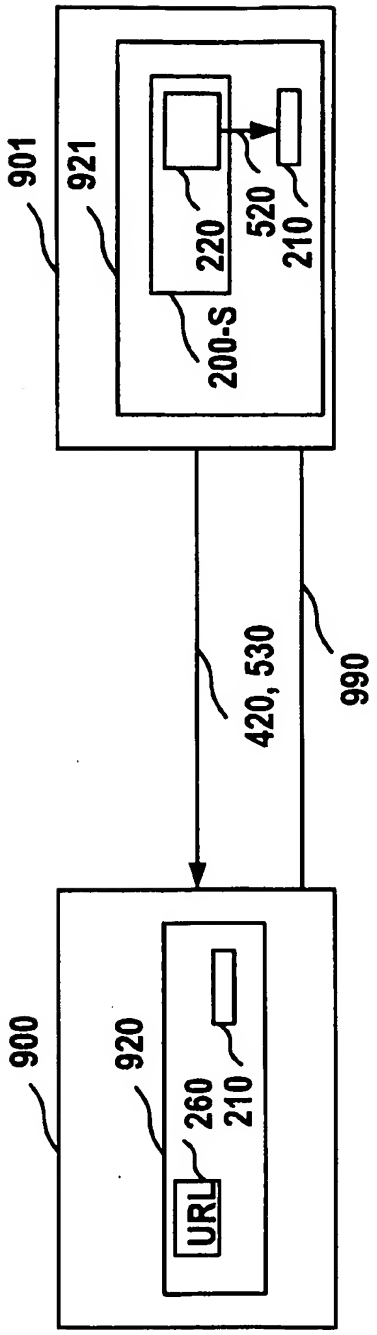
901
921
220
200-S
520
210
420, 530
990
900
920
260
210
URL

FIG. 3B

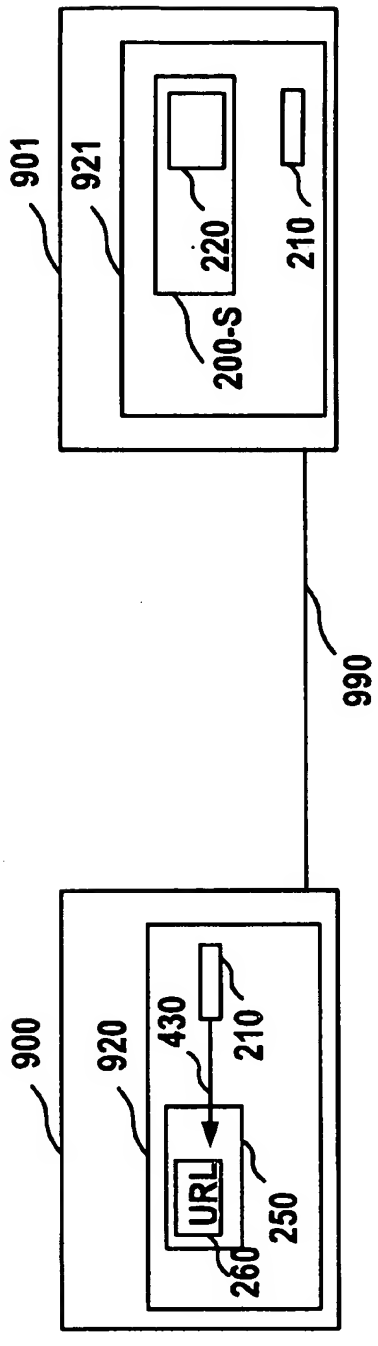

FIG. 3C

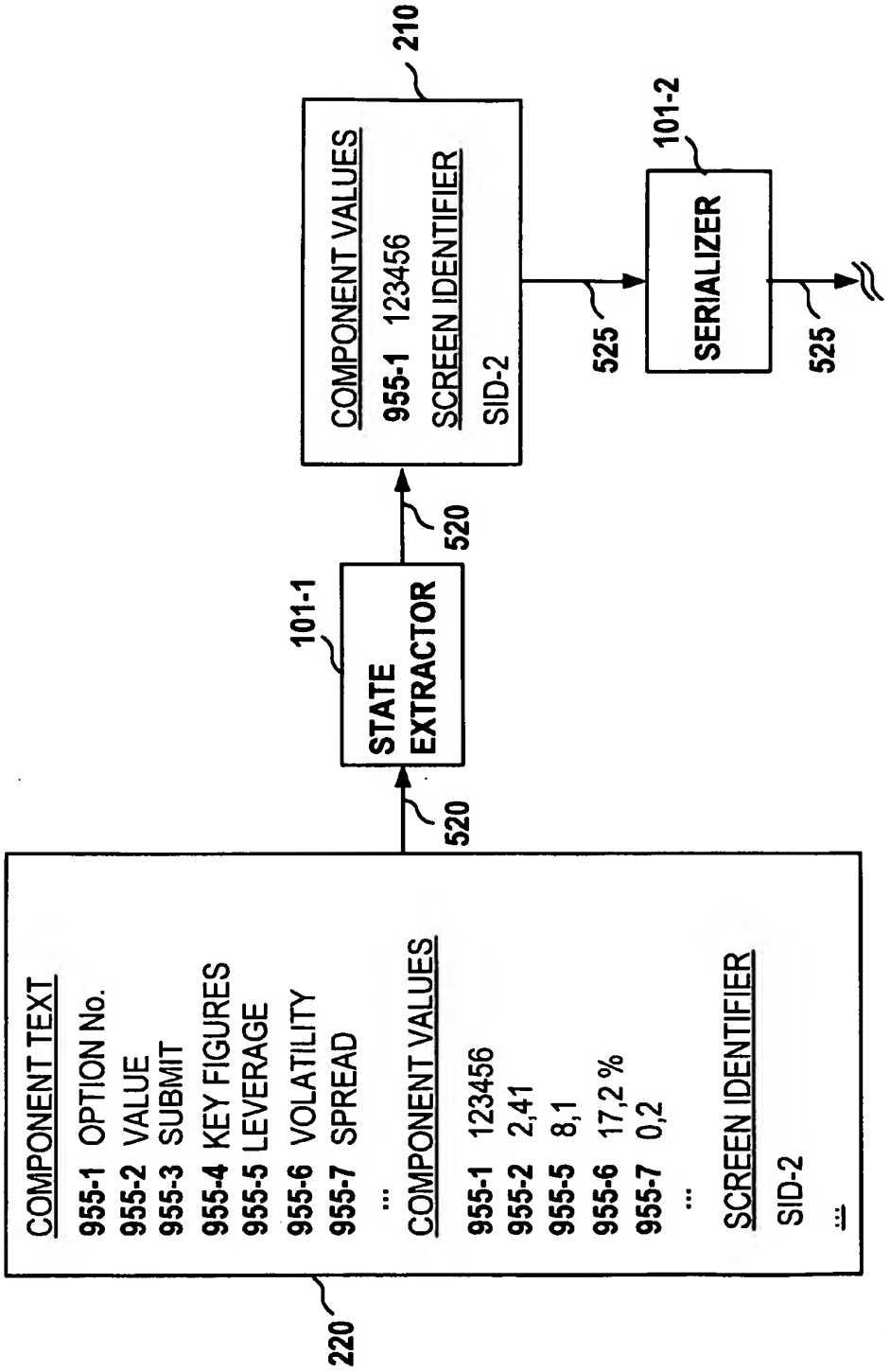
COMPONENT TEXT
955-1 OPTION No.
955-2 VALUE
955-3 SUBMIT
955-4 KEY FIGURES
955-5 LEVERAGE
955-6 VOLATILITY
955-7 SPREAD
...
COMPONENT VALUES
955-1 123456
955-2 2,41
955-5 8,1
955-6 17,2 %
955-7 0,2
...
SCREEN IDENTIFIER
SID-2
...
220
520
101-1
STATE EXTRACTOR
520
210
COMPONENT VALUES
955-1 123456
SCREEN IDENTIFIER
SID-2
525
101-2
SERIALIZER
525

FIG. 4

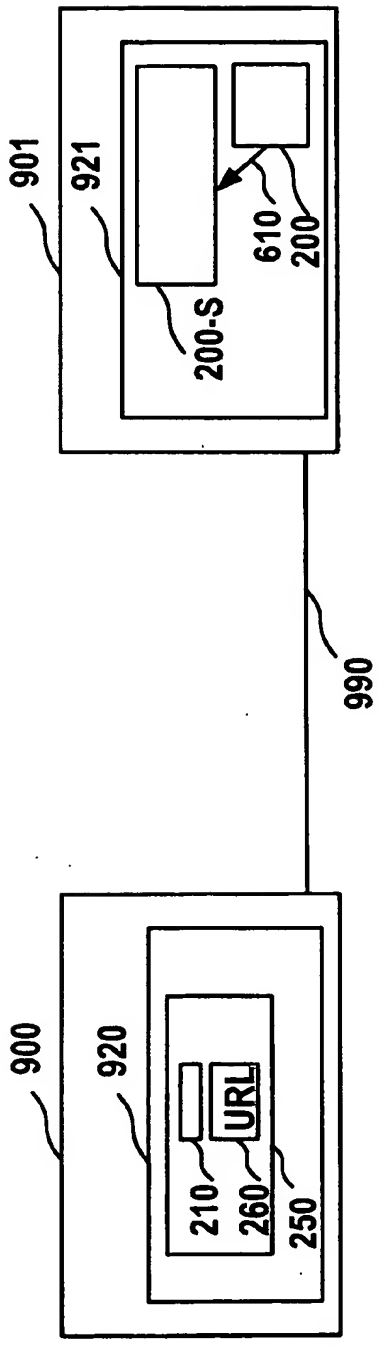
901
921
610
200
200-S
990
900
920
URL
210
260
250

FIG. 5A

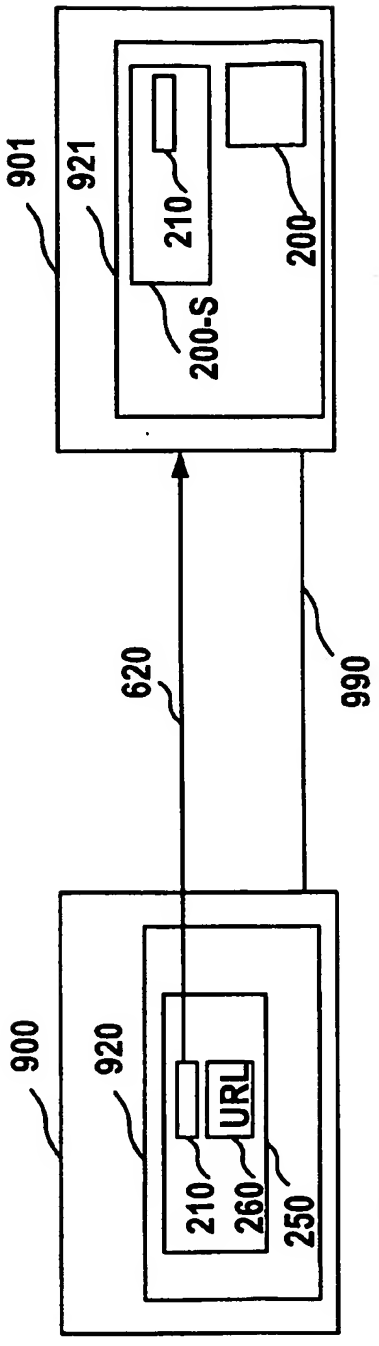
901
921
210
200
200-S
620
990
900
920
URL
210
260
250

FIG. 5B

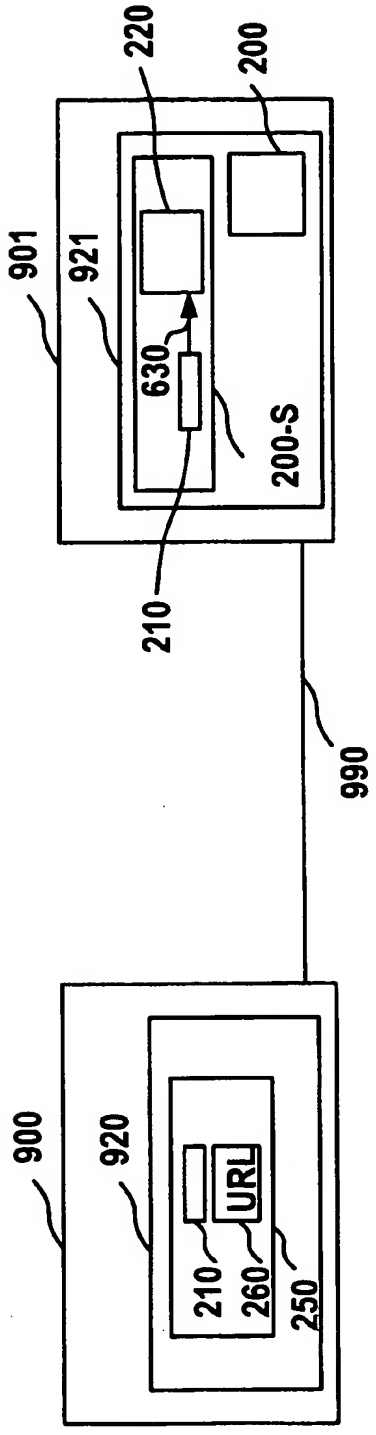

FIG. 5C

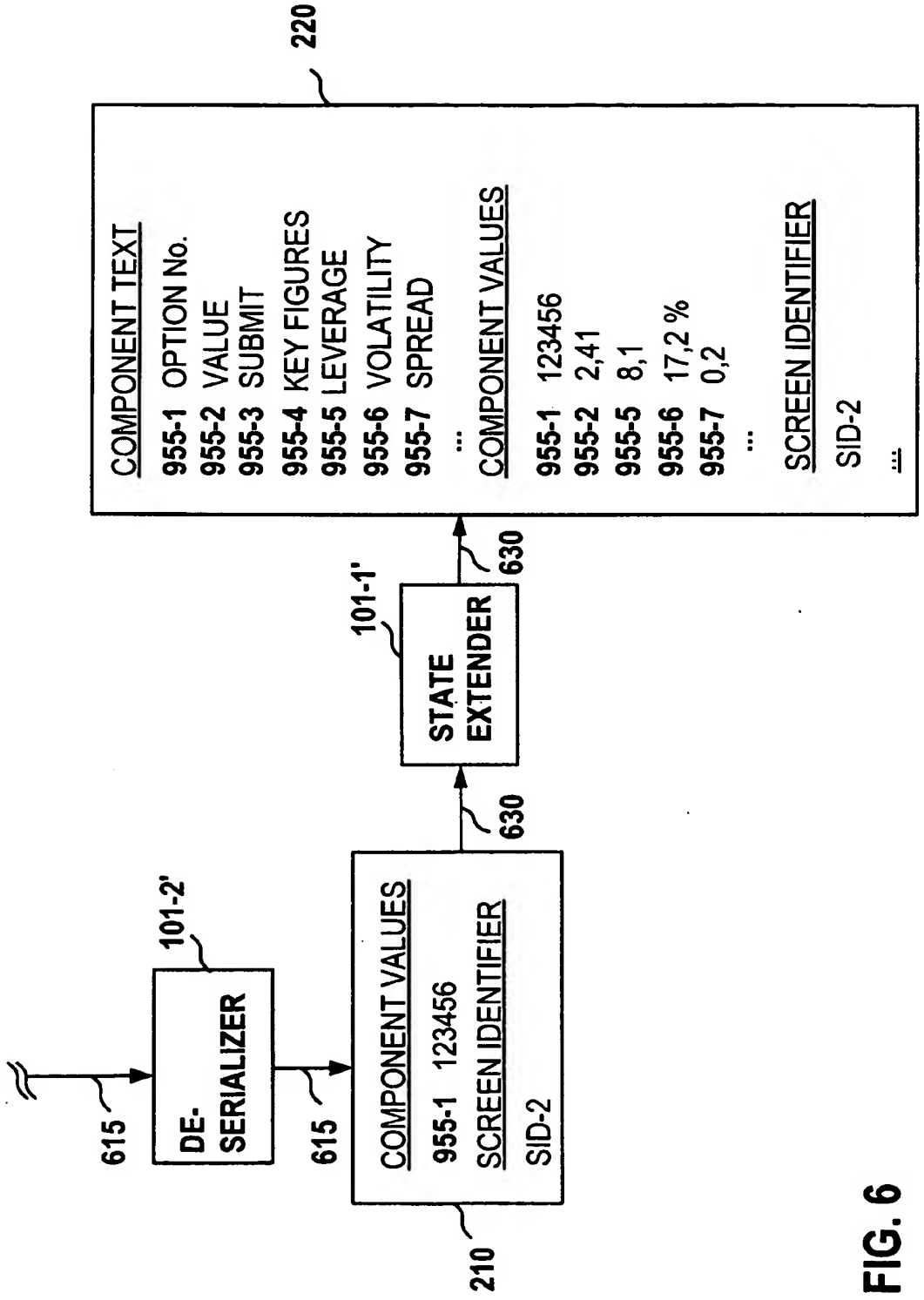
220
COMPONENT TEXT
955-1 OPTION No.
955-2 VALUE
955-3 SUBMIT
955-4 KEY FIGURES
955-5 LEVERAGE
955-6 VOLATILITY
955-7 SPREAD
...
COMPONENT VALUES
955-1 123456
955-2 2,41
955-5 8,1
955-6 17,2 %
955-7 0,2
...
SCREEN IDENTIFIER
SID-2
...
101-1'
STATE EXTENDER
630
630
101-2'
DE-SERIALIZER
615
615
COMPONENT VALUES
955-1 123456
SCREEN IDENTIFIER
SID-2
210

FIG. 6

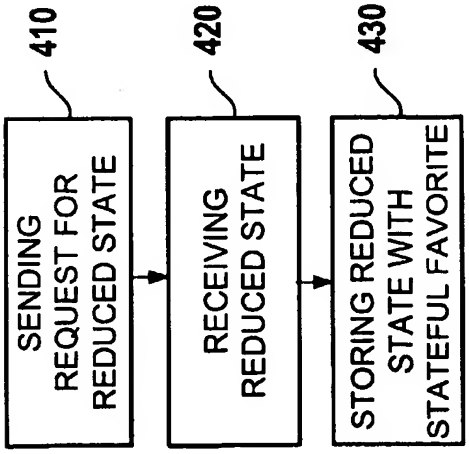

FIG. 7A

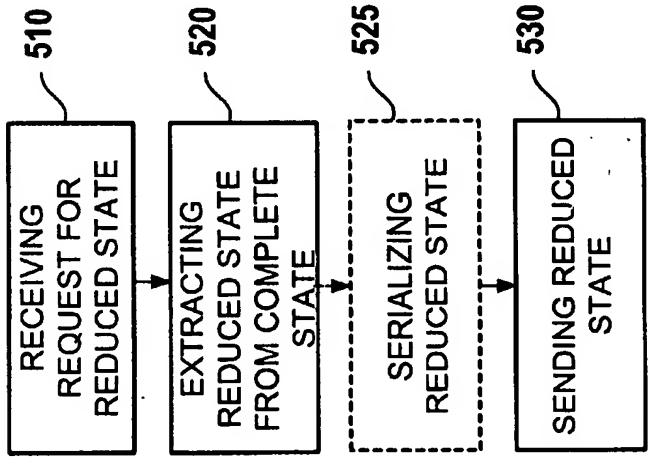

FIG. 7B

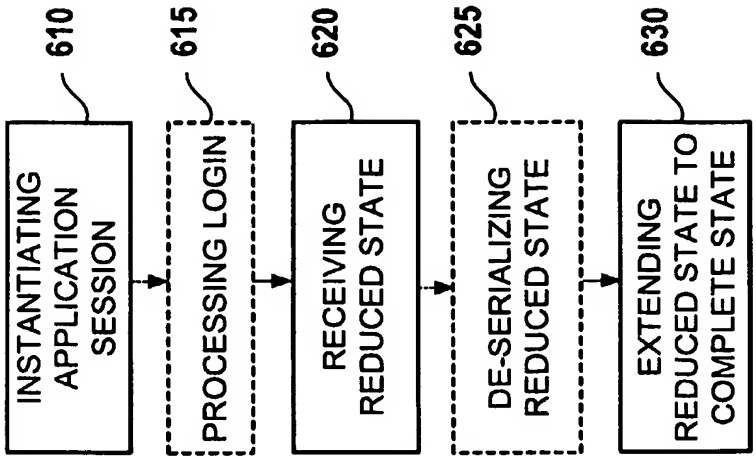

FIG. 7C

European Patent Office

EUROPEAN SEARCH REPORT

Application Number
EP 02 00 9955

DOCUMENTS CONSIDERED TO BE RELEVANT

Category	Citation of document with indication, where appropriate, of relevant passages	Relevant to claim	CLASSIFICATION OF THE APPLICATION (Int.Cl.7)
A	EP 0 938 045 A (IMEC VZW) 25 August 1999 (1999-08-25) * page 3, line 1 - page 5, line 5 *	1-24	G06F9/44
A	GOSCHKA K M ET AL: "Dynamic hyperlink generation for navigation in relational databases" 1999, NEW YORK, NY, USA, ACM, USA, February 1999 (1999-02), pages 23-24, XP002265044 ISBN: 1-58113-064-3 * the whole document *	1-24	

TECHNICAL FIELDS SEARCHED (Int.Cl.7)

G06F

The present search report has been drawn up for all claims

Place of search	Date of completion of the search	Examiner
The Hague	15 December 2003	Brandt, J

CATEGORY OF CITED DOCUMENTS

X : particularly relevant if taken alone
Y : particularly relevant if combined with another document of the same category
A : technological background
O : non-written disclosure
P : intermediate document

T : theory or principle underlying the invention
E : earlier patent document, but published on, or after the filing date
D : document cited in the application
L : document cited for other reasons

& : member of the same patent family, corresponding document

EPO FORM 1503 03.82 (P04C01)

ANNEX TO THE EUROPEAN SEARCH REPORT ON EUROPEAN PATENT APPLICATION NO. EP 02 00 9955

This annex lists the patent family members relating to the patent documents cited in the above-mentioned European search report. The members are as contained in the European Patent Office EDP file on
The European Patent Office is in no way liable for these particulars which are merely given for the purpose of information.

15-12-2003

Patent document cited in search report	Publication date	Patent family member(s)	Publication date
EP 0938045 A	25-08-1999	EP 0938045 A1	25-08-1999

EPO FORM P0459

For more details about this annex : see Official Journal of the European Patent Office, No. 12/82